TB-0356

Secure Shell Availability on the Mainframe

Issue Date: December 22, 2004
Effective Date: January 3, 2005
Section/Group: Systems Programming
Submitted by: Michael Knorr
Approved by: Clair Christensen

OpenSSH, or Secure Shell, which provides secure encryption for both remote login and file transfer, is now available on the State of Utah's IBM z/OS mainframe computers. It is intended to allow a non-z/OS server to communicate securely with a z/OS server using the ssh protocol in a security rich environment.

Utilities that it includes are:

> **ssh**—A client program for logging into a remote host and for executing commands on a remote host. It is an alternative to rlogin and rsh, and provides secure encrypted communications between two untrusted hosts over an insecure network.
> **sftp**—For file transfers over an encrypted ssh transport. It is an interactive file transfer program similar to ftp. It uses such features of ssh as public key authentication and compression.
> **scp**—For copying files between hosts in a network. It is an alternative to rcp. It also uses public key authentication and provides the same security as ssh.

Other basic utilities, such as ssh-keygen and ssh-keyscan, are also included.

Of its authentication methods, OpenSSH for z/OS supports RSA-based authentication based on public-key cryptography. A password is not used nor asked for to log in to the remote host. The State of Utah utilizes SSH protocol version 2, which allows the RSA or DSA algorithm to be used. Protocol 2 also provides additional mechanisms for confidentiality (the traffic is encrypted using 3DES, Blowfish, CAST128, or Arcfour) and integrity (hmac-md5, hmac-sha1).

If public-key authentication fails, ssh prompts the user for a password which is sent to the remote host for checking. However, because all communications are encrypted, the password cannot be seen by anyone listening on the network.

OpenSSH also supports X11 forwarding. X11 forwarding allows users who have an account on a UNIX machine to open a connection to the X11 interface remotely from another machine. However, X11 forwarding has not been configured on the State of Utah's mainframes. If you require X11 forwarding, please contact a UNIX System Services administrator.

Default system installation of SSH on the z/OS mainframe servers is complete. The following describes the setup tasks in z/OS UNIX System Services (omvs) an SSH user must perform:

In the ISPF Command Shell, issue command `omvs` to enter UNIX System Services.

Find your $HOME directory — `echo $HOME`.

Create file `$HOME/.profile`

Edit `$HOME/.profile` and add record:

```
      export STEPLIB=none
```

```
mkdir $HOME/.ssh ; chmod 700 $HOME/.ssh
```

If you wish to customize your own ssh client-side configuration file:

```
      cp -p /samples/ssh_config $HOME/.ssh/config
```

If you do not, the default `/etc/ssh/ssh_config` will be used. See Note 1 below.

Create your `authorized_keys` file. Issue commands:

```
      touch $HOME/.ssh/authorized_keys ;
      chmod 700 $HOME/.ssh/authorized_keys
```

The resulting permissions for `authorized_keys` ***must*** be 700.

Generate user authentication keys:

```
      ssh-keygen -t dsa
      ssh-keygen -t rsa
```

Respond to subsequent prompts, making sure keys are stored in your `$HOME/.ssh` directory; skip password prompts by pressing the Enter key.

You should see the following pairs of private and public keys in your `$HOME/.ssh` directory:

```
id_dsa
id_dsa.pub
id_rsa
id_rsa.pub
```

Ftp (or equivalent) the resulting public keys to all remote hosts you plan to log in to using public key authentication. Append the public keys to the remote user's `$HOME/.ssh/authorized_keys` file. Conversely, to enable the remote user to log in to your local account, append the remote user's public keys to your local `$HOME/.ssh/authorized_keys` file. If you do not enable remote users to log in to your local account, your `authorized_keys` file will remain empty.

To append keys, issue the command:

```
cat local.id_dsa.pub local.id_rsa.pub >> $HOME/.ssh/authorized_keys
```

on the remote machine.

The end result and location of keys is as follows:

Local System		**Remote System**
`$HOME/.ssh/id_dsa`		
`$HOME/.ssh/id_rsa`		
`$HOME/.ssh/id_dsa.pub`	→	`$HOME/.ssh/authorized_keys`
`$HOME/.ssh/id_rsa.pub`	→	`$HOME/.ssh/authorized_keys`

Every time you regenerate your keys, you must update the `authorized_keys` file on the remote system.

In your `$HOME/.ssh` directory, you should see the following files with the following attributes:

```
drwxr-xr-x        2 AOPSTART DP@OMVS    8192 Nov 19 15:09 .
drwxr-xr-x       22 AOPSTART DP@OMVS    8192 Nov 22 08:53 ..
-rwx------  --s-  1 AOPSTART DP@OMVS     828 Nov 19 09:55 authorized_keys
-rw-------  --s-  1 AOPSTART DP@OMVS     668 Nov 19 09:26 id_dsa
-rw-r--r--  --s-  1 AOPSTART DP@OMVS     604 Nov 19 09:26 id_dsa.pub
-rw-------  --s-  1 AOPSTART DP@OMVS     883 Nov 19 09:20 id_rsa
-rw-r--r--  --s-  1 AOPSTART DP@OMVS     224 Nov 19 09:20 id_rsa.pub
-rw-r--r--  --s-  1 AOPSTART DP@OMVS    1233 Nov 19 14:38 config
```

To log in to the remote system using your configuration file found at `$HOME/.ssh/config` issue:

```
ssh -F $HOME/.ssh/config user@hostname
```

To log in to the remote system using the default configuration file found at `/etc/ssh/ssh_config` issue:

```
ssh user@hostname
```

To Secure Shell ftp into the remote system, substitute the `sftp` command for the `ssh` command in the above two examples.

Note 1: The default configuration file `/etc/ssh/ssh_config` expects to find user private keys at `~/.ssh/id_dsa` and at `~/.ssh/id_rsa`. If you place `id_dsa` and `id_rsa` in any other directory path you must use your own configuration file, specifying the directory path followed by the private key file name as a parameter of the `IdentityFile` keyword, or you must execute `ssh` with the "`-i identity_file`" option. In addition, `/etc/ssh/ssh_config` also expects to find the user `known_hosts` file at `~/.ssh/known_hosts`. If you place `known_hosts` in any other directory path you must use your own configuration file, specifying the directory path followed by the `known_hosts` file name as a parameter of the `UserKnownHostsFile` keyword.

Note 2: If you have performed the above steps correctly, you should be able to log in to a remote host without the need for a user ID and/or a password. If, however, you are prompted for a user ID and/or a password or passphrase, you have not completed all of the above steps correctly